Opérateurs arithmétiques matériels
Systèmes à plusieurs bases

Florent de Dinechin

13 décembre 2005

Double base

- Idée :

$$X = \sum_{i,j} d_{i,j} \beta^i \gamma^j \quad d_{i,j} \in \{0, 1\}$$

Double base

- Idée :

$$X = \sum_{i,j} d_{i,j} \beta^i \gamma^j \quad d_{i,j} \in \{0, 1\}$$

- (Bien sûr on pourra généraliser à plus de bases et à des chiffres pas binaires)

Double base

- Idée :

$$X = \sum_{i,j} d_{i,j} \beta^i \gamma^j \quad d_{i,j} \in \{0, 1\}$$

- (Bien sûr on pourra généraliser à plus de bases et à des chiffres pas binaires)
- Prenons pour commencer $\beta = 2$ et $\gamma = 3$.

Double base

- Idée :

$$X = \sum_{i,j} d_{i,j} \beta^i \gamma^j \quad d_{i,j} \in \{0, 1\}$$

- (Bien sûr on pourra généraliser à plus de bases et à des chiffres pas binaires)
- Prenons pour commencer $\beta = 2$ et $\gamma = 3$.
- Deux approches ont été étudiées un peu en détail :
  - Manipuler des nombres comme de petites listes de termes $2^i 3^j$, i et j étant des entiers relatifs (codés sur peu de bits) (DBNS),

Double base

- Idée :
$$X = \sum_{i,j} d_{i,j} \beta^i \gamma^j \quad d_{i,j} \in \{0, 1\}$$
- (Bien sûr on pourra généraliser à plus de bases et à des chiffres pas binaires)
- Prenons pour commencer $\beta = 2$ et $\gamma = 3$.
- Deux approches ont été étudiées un peu en détail :
  - Manipuler des nombres comme de petites listes de termes $2^i 3^j$, i et j étant des entiers relatifs (codés sur peu de bits) (DBNS),
  - Manipuler des réels comme un seul terme : $X = 2^i 3^j$, i et j étant des rationnels codés en virgule fixe (DBLNS)

Double base

- Idée :

$$X = \sum_{i,j} d_{i,j} \beta^i \gamma^j \quad d_{i,j} \in \{0, 1\}$$

- (Bien sûr on pourra généraliser à plus de bases et à des chiffres pas binaires)
- Prenons pour commencer $\beta = 2$ et $\gamma = 3$.
- Deux approches ont été étudiées un peu en détail :
  - Manipuler des nombres comme de petites listes de termes $2^i 3^j$, i et j étant des entiers relatifs (codés sur peu de bits) (DBNS),
  - Manipuler des réels comme un seul terme : $X = 2^i 3^j$, i et j étant des rationnels codés en virgule fixe (DBLNS)

Vassil S. Dimitrov, Graham A. Jullien, and William C. Miller, *Theory and Applications of the Double-Base Number System,* IEEE Transactions on Computers, vol. 48, no. 10, october 1999

DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Représentation géométrique :
  - damier avec i en abscisse, j en ordonnée
  - une croix noire si $d_{i,j} = 1$

DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Représentation géométrique :
  - damier avec i en abscisse, j en ordonnée
  - une croix noire si $d_{i,j} = 1$
- Représentation très creuse :
  - 23 is the smallest integer requiring three 2-integers.

DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Représentation géométrique :
  - damier avec i en abscisse, j en ordonnée
  - une croix noire si $d_{i,j} = 1$
- Représentation très creuse :
  - 23 is the smallest integer requiring three 2-integers.
  - The smallest integer requiring four 2-integers is 431,

DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0,1\}$$

- Représentation géométrique :
  - damier avec i en abscisse, j en ordonnée
  - une croix noire si $d_{i,j} = 1$
- Représentation très creuse :
  - 23 is the smallest integer requiring three 2-integers.
  - The smallest integer requiring four 2-integers is 431,
  - five 2-integers are needed to represent 18,431

DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Représentation géométrique :
  - damier avec i en abscisse, j en ordonnée
  - une croix noire si $d_{i,j} = 1$
- Représentation très creuse :
  - 23 is the smallest integer requiring three 2-integers.
  - The smallest integer requiring four 2-integers is 431,
  - five 2-integers are needed to represent 18,431
  - and six 2-integers are needed to represent 3,448,733. Up to this limit we can represent every integer as a sum of at most five 2-integers.

DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Représentation géométrique :
  - damier avec i en abscisse, j en ordonnée
  - une croix noire si $d_{i,j} = 1$
- Représentation très creuse :
  - 23 is the smallest integer requiring three 2-integers.
  - The smallest integer requiring four 2-integers is 431,
  - five 2-integers are needed to represent 18,431
  - and six 2-integers are needed to represent 3,448,733. Up to this limit we can represent every integer as a sum of at most five 2-integers.
- Représentation très redondante (d'habitude on aime).

Conversions

- Conversion de DBNS en binaire facile (comment ?)

Conversions

- Conversion de DBNS en binaire facile (comment ?)
- Dans l'autre sens : trouver la plus petite représentation ("CDBNR" pour *canonical*) est sans doute NP-tout

Conversions

- Conversion de DBNS en binaire facile (comment ?)
- Dans l'autre sens : trouver la plus petite représentation ("CDBNR" pour *canonical*) est sans doute NP-tout
- Mais l'heuristique gloutonne marche bien ("NCDBNR" pour *near canonical*)

Conversions

- Conversion de DBNS en binaire facile (comment ?)
- Dans l'autre sens : trouver la plus petite représentation ("CDBNR" pour *canonical*) est sans doute NP-tout
- Mais l'heuristique gloutonne marche bien ("NCDBNR" pour *near canonical*)
  - on commence par chercher le plus grand $2^i 3^j$ inférieur à X
  - on le retranche à X et on recommence avec le reste

Conversions

- Conversion de DBNS en binaire facile (comment ?)
- Dans l'autre sens : trouver la plus petite représentation (“CDBNR” pour *canonical*) est sans doute NP-tout
- Mais l'heuristique gloutonne marche bien (“NCDBNR” pour *near canonical*)
  - on commence par chercher le plus grand $2^i 3^j$ inférieur à X
  - on le retranche à X et on recommence avec le reste

Ces gens aiment bien les sigles

Identités remarquables (redondance)

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

Sur le dessin :

- réduction de ligne par $1 + 2 = 3$

Identités remarquables (redondance)

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

Sur le dessin :

- réduction de ligne par $1 + 2 = 3$ et $8 + 1 = 9$

Identités remarquables (redondance)

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

Sur le dessin :

- réduction de ligne par $1 + 2 = 3$ et $8 + 1 = 9$
- réduction de colonne par $3 + 1 = 4$

Un générateur d'identités remarquables

Généralisation par des équations diophantiennes

- Theorem 2. The Diophantine equation $x + y = z$, where $\mathrm{GCD}(x, y, z) = 1$ and x, y, and z are 6-integers of the form $2^i 3^j 5^k 7^l 11^m 13^n$ where $i, j, k, l, m, n \in \mathbf{N}$ has exactly 545 solutions.

Un générateur d'identités remarquables

Généralisation par des équations diophantiennes

- Theorem 2. The Diophantine equation $x + y = z$, where $\mathrm{GCD}(x, y, z) = 1$ and x, y, and z are 6-integers of the form $2^i 3^j 5^k 7^l 11^m 13^n$ where $i, j, k, l, m, n \in \mathbf{N}$ has exactly 545 solutions.
- En fixant $k = l = m = n = 0$ il reste trois solutions :
  - $1 + 2 = 3$
  - $1 + 3 = 4$
  - $1 + 8 = 9$

Un générateur d'identités remarquables

Généralisation par des équations diophantiennes

- Theorem 2. The Diophantine equation $x + y = z$, where $\mathrm{GCD}(x, y, z) = 1$ and x, y, and z are 6-integers of the form $2^i 3^j 5^k 7^l 11^m 13^n$ where $i, j, k, l, m, n \in \mathbf{N}$ has exactly 545 solutions.
- En fixant $k = l = m = n = 0$ il reste trois solutions :
  - $1 + 2 = 3$
  - $1 + 3 = 4$
  - $1 + 8 = 9$
- Compactions 3 en 1 : il existe 27 solutions à $x + y + z = t$, par exemple $1 + 216 + 512 = 729$

Un générateur d'identités remarquables

Généralisation par des équations diophantiennes

- Theorem 2. The Diophantine equation $x + y = z$, where GCD$(x, y, z) = 1$ and x, y, and z are 6-integers of the form $2^i 3^j 5^k 7^l 11^m 13^n$ where $i, j, k, l, m, n \in \mathbf{N}$ has exactly 545 solutions.
- En fixant $k = l = m = n = 0$ il reste trois solutions :
  - $1 + 2 = 3$
  - $1 + 3 = 4$
  - $1 + 8 = 9$
- Compactions 3 en 1 : il existe 27 solutions à $x + y + z = t$, par exemple $1 + 216 + 512 = 729$
- Bien sûr cela permet de compacter une représentation DBNS

Un générateur d'identités remarquables

Généralisation par des équations diophantiennes

- Theorem 2. The Diophantine equation $x + y = z$, where $\mathrm{GCD}(x, y, z) = 1$ and x, y, and z are 6-integers of the form $2^i 3^j 5^k 7^l 11^m 13^n$ where $i, j, k, l, m, n \in \mathbf{N}$ has exactly 545 solutions.
- En fixant $k = l = m = n = 0$ il reste trois solutions :
  - $1 + 2 = 3$
  - $1 + 3 = 4$
  - $1 + 8 = 9$
- Compactions 3 en 1 : il existe 27 solutions à $x + y + z = t$, par exemple $1 + 216 + 512 = 729$
- Bien sûr cela permet de compacter une représentation DBNS
- Ils sont bien allusifs sur l'ordre d'application de ces opérations (matériel ?)

Addition DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Idée : on fusionne les listes de termes

Addition DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Idée : on fusionne les listes de termes
- Puis on (pseudo) normalise la somme par les identités remarquables

Addition DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Idée : on fusionne les listes de termes
- Puis on (pseudo) normalise la somme par les identités remarquables
- Pb : “propagations de retenue” verticales (dessin)

Addition DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Idée : on fusionne les listes de termes
- Puis on (pseudo) normalise la somme par les identités remarquables
- Pb : "propagations de retenue" verticales (dessin)
- Comme il y a $n/\log_2(3)$ lignes par rapport à une représentation binaire cela fait 37% de propagation de retenues en moins

Addition DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0,1\}$$

- Idée : on fusionne les listes de termes
- Puis on (pseudo) normalise la somme par les identités remarquables
- Pb : “propagations de retenue” verticales (dessin)
- Comme il y a $n/\log_2(3)$ lignes par rapport à une représentation binaire cela fait 37% de propagation de retenues en moins
- Pour que la fusion se fasse sans douleur, on préprocesse les deux termes pour les mettre sous la forme “ARDBNR” *(addition-ready)*
  - il n'y a pas de termes de la forme $2^i 3^j + 2^{i+1} 3^j$
  - (on les a remplacés par $2^i 3^{j+1}$)

Addition DBNS

$$X = \sum_{i,j} d_{i,j} 2^i 3^j \quad d_{i,j} \in \{0, 1\}$$

- Idée : on fusionne les listes de termes
- Puis on (pseudo) normalise la somme par les identités remarquables
- Pb : “propagations de retenue” verticales (dessin)
- Comme il y a $n/\log_2(3)$ lignes par rapport à une représentation binaire cela fait 37% de propagation de retenues en moins
- Pour que la fusion se fasse sans douleur, on préprocesse les deux termes pour les mettre sous la forme “ARDBNR” *(addition-ready)*
  - il n'y a pas de termes de la forme $2^i 3^j + 2^{i+1} 3^j$
  - (on les a remplacés par $2^i 3^{j+1}$)
  - alors si un terme $2^i 3^j$ apparait dans X et dans Y, on sait qu'on peut les remplacer dans la somme par $2^i 3^{j+1}$ qui n'y est pas.

Multiplication DBNS

- ...
- On peut l'exprimer comme des décalages dans le plan et des additions
- Faible densité de la représentation $\longrightarrow$ peu d'opérations
- Disent-ils

Ma conclusion sur DBNS

- Tout cela reste vague quant aux compromis d'implémentation

Ma conclusion sur DBNS

- Tout cela reste vague quant aux compromis d'implémentation
- A priori, un bon DBNS est un DBNS asynchrone (on manipule des listes)

Ma conclusion sur DBNS

- Tout cela reste vague quant aux compromis d'implémentation
- A priori, un bon DBNS est un DBNS asynchrone (on manipule des listes)
- Pas d'implémentation connue à part un vague filtre DSP en 2001

Ma conclusion sur DBNS

- Tout cela reste vague quant aux compromis d'implémentation
- A priori, un bon DBNS est un DBNS asynchrone (on manipule des listes)
- Pas d'implémentation connue à part un vague filtre DSP en 2001
- Utilité pour d'autres problèmes ?
  - J'ai regardé la multiplication par une constante

Ma conclusion sur DBNS

- Tout cela reste vague quant aux compromis d'implémentation
- A priori, un bon DBNS est un DBNS asynchrone (on manipule des listes)
- Pas d'implémentation connue à part un vague filtre DSP en 2001
- Utilité pour d'autres problèmes ?
  - J'ai regardé la multiplication par une constante
  - rappel : mult par une (grande) constante par additions et décalages
  - idée : si on sait écrire la constante dans un MDNS à base de (2, 3, 5, 9, 17...) on a tout de suite les décalages
  - et donc on veut une représentation la plus compacte possible.

Ma conclusion sur DBNS

- Tout cela reste vague quant aux compromis d'implémentation
- A priori, un bon DBNS est un DBNS asynchrone (on manipule des listes)
- Pas d'implémentation connue à part un vague filtre DSP en 2001
- Utilité pour d'autres problèmes ?
  - J'ai regardé la multiplication par une constante
  - rappel : mult par une (grande) constante par additions et décalages
  - idée : si on sait écrire la constante dans un MDNS à base de (2, 3, 5, 9, 17...) on a tout de suite les décalages
  - et donc on veut une représentation la plus compacte possible.
  - ... ben je n'ai même pas écrit un rapport de recherche :)

DBLNS ou MDLNS (ou DBNS à un chiffre)

- au tableau
- Un papier complet rien que sur la conversion de binaire vers MDLNS
- Le seul intérêt c'est de remplacer une addition n bits par deux additions $n/2$ dans la multiplication LNS, qui était l'opération facile.
- L'addition est sans espoir
- On perd la comparaison, mais elle n'est pas difficile à retrouver
- On perd l'élégance du LNS

Dimitrov et al, The Use of the Multi-Dimensional Logarithmic Number System in DSP Applications (Arith15)
Ce papier fait une somme de produits : les produits sont faits en MDLNS, l'accumulation en binaire à Papa.

Oh, une application

Dimitrov et al, The Use of the Multi-Dimensional Logarithmic Number System in DSP Applications (Arith15)

- somme de produits :
  - les produits sont faits en MDLNS,
  - l'accumulation en binaire à Papa.
- Travailler avec peu de bits d'exposant
- Pour éviter les overflow sur les exposants, prémultiplier par une bonne approx de 1 qui réduit les exposants susceptibles de déborder

Conclusion

Pour les gens de ma génération, il devrait être possible de se faire un nom sans avoir à inventer un système fantaisiste de représentation des nombres...